Meeting Minutes 28 March 2001

More Metadata

Author:
Melanie Nelson (Physiome Sciences Inc.)
Contributor:
Warren Hedley (Bioengineering Research Group, University of Auckland)

1 Introduction

While researching and writing up the correct usage of Dublin Core qualifiers in CellML metadata, Melanie began to think about how to use the Dublin Core qualifiers' elegant scheme of allowing a type, encoding scheme, and value for each type of metadata to solve remaining issues with biological entity, mathematical problem type, and annotation metadata. These meeting minutes discuss her proposals for each of these types of metadata and the resolution of the questions about whether or not to include the Dublin Core description, contributor, and relation metadata in the recommended set of CellML metadata. The Dublin Core qualifiers are introduced in the March 26 meeting minutes[1].

2 Biological Entity Metadata

2.1 Background

The March 15 meeting minutes[2] indicate that the remaining issue with biological entity metadata is how to refer to biological entities by a database unique identifier. This requires referring to both the identity of the database and the value of the unique identifier for the entity within the database.

This is a perfect opportunity to use an encoding scheme/value pair. It must also be possible to refer to a biological entity by name. This led to a system whereby "name" was considered another encoding scheme, resulting in the RDF shown in Figure 1.

This implementation is not adequate, because it is difficult to provide both a name and a database unique identifier for a biological entity. The `<rdf:Alt>` container was considered for this purpose. However, due to the requirement that the value enclosed in an `<rdf:li>` element be either a string or an `<rdf:Description>` element (see the BNF on page 17 of the RDF Model and Syntax Specification[3], the use of the `<rdf:Alt>` container produces unnecessarily verbose RDF, as shown in Figure 2.

The complete formal grammar for RDF (presented in section 6 of the RDF specification) includes a BNF statement that seems to indicate that any *typed node* (which seems to be any valid element) may be considered a "description". However, since the RDF specification is unclear on this, but is clear on the fact that use of the `<rdf:Description>` element is the recommended practice, it would probably be unwise to omit the `<rdf:Description>` elements inside the `<rdf:li>` elements in this example.

There are two problems with this implementation (in addition to its verbosity):

- It obscures the fact the name of the biological entity is really just a title that can be represented with the Dublin Core `<title>` element.

[1] http://www.cellml.org/private/progress_reports/20010326_meeting_minutes.html
[2] http://www.cellml.org/private/progress_reports/20010315_meeting_minutes.html
[3] http://www.w3.org/TR/REC-rdf-syntax/

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cmeta="http://www.cellml.org/2001/03/metadata#">

  <rdf:Description about="some_element_id">
    <cmeta:bio_entity>
      <rdf:Description>
        <cmeta:entity_scheme>name</cmeta:entity_scheme>
        <rdf:value>HERG</rdf:value>
      </rdf:Description>
    </cmeta:bio_entity>
  </rdf:Description>
</rdf:RDF>
```

FIGURE 1: Preliminary implementation for biological entity metadata, in which “name” is treated as an encoding scheme.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cmeta="http://www.cellml.org/2001/03/metadata#">

  <rdf:Description about="some_element_id">
    <cmeta:bio_entity>
        <rdf:Alt>
          <rdf:li>
            <rdf:Description>
              <cmeta:entity_scheme>name</cmeta:entity_scheme>
              <rdf:value>calmodulin</rdf:value>
            </rdf:Description>
          </rdf:li>
          <rdf:li>
            <rdf:Description>
              <cmeta:entity_scheme>SWISS-PROT</cmeta:entity_scheme>
              <rdf:value>CALM_HUMAN</rdf:value>
            </rdf:Description>
          </rdf:li>
        </rdf:Alt>
    </cmeta:bio_entity>
  </rdf:Description>
</rdf:RDF>
```

FIGURE 2: A possible implementation for biological entity metadata using the `<rdf:Alt>` container element.

- The ability to specify multiple database entries for a given biological entity is not required by the CellML requirements. Furthermore, allowing multiple database entries increases the risk of errors, because it would allow for inconsistent database entires to be specified. While it is not essential that CellML's metadata system prevent such errors, it is desirable to do so whenever possible.

Reflection on these two points led to the implementation presented in the next section.

2.2 Melanie's Initial Implementation

Figure 3 shows another implementation for biological entity metadata. This system allows a name and a database unique identifier to be stored for a biological entity.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cmeta="http://www.cellml.org/2001/03/metadata#"
    xmlns:dc="http://purl.org/dc/elements/1.0/">

  <rdf:Description about="some_element_id">
    <cmeta:bio_entity>
      <rdf:Description>
        <dc:title>calmodulin</dc:title>
        <cmeta:entity_scheme>SWISS-PROT</cmeta:entity_scheme>
        <rdf:value>CALM_HUMAN</rdf:value>
      </rdf:Description>
    </cmeta:bio_entity>
  </rdf:Description>
</rdf:RDF>
```

FIGURE 3: Yet another implementation for biological entity metadata.

Warren was concerned that we should provide the ability for modellers to define more than one database identifier for a given biological entity. To make sure that software would be able to adequately deal with inconsistent identifiers, it was suggested that an `<rdf:Alt>` element be used to group together these identifiers, indicating that software could only use one of the alternatives on offer. A valid CellML document would have to indicate which database identifier was the primary identifier by specifying that all other identifiers were of type `alternative`.

When the verbosity of this solution became apparent (when forced to contemplate the markup/information ratio in examples such as that in Figure 5), Warren had a strong allergic reaction. He hated it so much that he went temporarily insane and considered ditching RDF altogether. Melanie talked him out of this idea, and together they came up with the recommended implementation presented in the next section.

2.3 Recommended Implementation

Figure 4 shows the recommended implementation for biological entity metadata in CellML. In this implementation, the name of the biological entity is handled exactly as alternative names for CellML elements are handled (with the `<dc:title>` element, which may be qualified by a `<dcq:titleType>` element). Multiple database identifiers may be provided. Each one is stored in a `<cmeta:identifier>` element, which must be qualified by a `<cmeta:identifier_scheme>` element that identifies the database. The CellML metadata specification will define names for certain encoding schemes (see below). The `<cmeta:identifier>` element may also be qualified by a `<cmeta:identifier_type>` element. This element should have a value of “alternative” for all `<cmeta:identifier>` elements except for one, which is considered the primary identifier. This addresses the concern about allowing multiple database identifiers, which might actually refer to different biological entities. This error may still occur, but now software has a method by which to determine which identifier should be given precedence.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cmeta="http://www.cellml.org/2001/03/metadata#"
    xmlns:dc="http://purl.org/dc/elements/1.0/">

  <rdf:Description about="some_element_id">
    <cmeta:bio_entity>
      <rdf:Description>
        <dc:title>calmodulin</dc:title>
        <cmeta:identifier>
          <rdf:Description>
            <cmeta:identifier_scheme>SWISS-PROT</cmeta:identifier_scheme>
            <rdf:value>CALM_HUMAN</rdf:value>
          </rdf:Description>
        </cmeta:identifier>
        <cmeta:identifier>
          <rdf:Description>
            <cmeta:identifier_scheme>GenBank</cmeta:identifier_scheme>
            <cmeta:identifier_type>alternative</cmeta:identifier_type>
            <rdf:value>P02593</rdf:value>
          </rdf:Description>
        </cmeta:identifier>
      </rdf:Description>
    </cmeta:bio_entity>
  </rdf:Description>

</rdf:RDF>
```

FIGURE 4: The recommended implementation for biological entity metadata.

The following points will be made in the metadata specification:

- Multiple names can be handled using the same method as is used for the alternative name metadata. Specifically, we can use the Dublin Core type qualifier on the `<dc:title>` element. This is shown in Figure 5. This implementation is chosen over the `<rdf:Alt>` container element to preserve consistency with the “Dublin Core in RDF” draft proposal (described in the March 26 meeting minutes).
- The CellML metadata specification will define the following encoding schemes:
  - SWISS-PROT (SWISS-PROT protein database)
  - GenBank (GenBank nucleic acid database)
  - webpage (webpage providing info about the biological entity)

  Model authors and authors of processing software are free to define additional encoding schemes. However, software claiming to be “CellML metadata compliant” is not required to recognise these schemes.

- RDF containers can be used to indicate that a given CellML element is relevant for more than one biological entity. An `<rdf:Bag>` element can be used to indicate that the CellML element is relevant for an entire group of biological entities. An `<rdf:Alt>` element can be used to indicate that the CellML element can be relevant for one member of a group of entities. Note that the first member listed in the `<rdf:Alt>` element will be considered the preferred value. The use of the `<rdf:Bag>` element is shown in Figure 5. The use of the `<rdf:Alt>` element would be identical. "CellML metadata compliant" software will be required to recognise RDF containers in biological entity metadata. The use of RDF containers is preferred to simply repeating the `<cmeta:bio_entity>` element because it removes all ambiguity about how the group of biological entities relates to the referenced CellML element. Furthermore, it is consistent with the recommendations of the "Dublin Core in RDF" draft proposal.

3 Mathematical Problem Type

3.1 Background

The March 15 meeting minutes[4] indicate that the remaining issue with mathematical problem type metadata is how to provide a URL for the encoding scheme that is used for the problem type. Furthermore, the implementation of the encoding scheme qualifier in this metadata needs to be made consistent with the metadata qualified with the Dublin Core qualifiers.

3.2 Recommendations

There is actually no need to provide a URL for the encoding scheme used for the problem type metadata. We can adopt a solution similar to that used for the biological entity encoding schemes. We will define the `GAMS` encoding scheme in the CellML metadata specification. Model authors and authors of CellML processing software are free to create their own encoding schemes. However, software that claims to be "CellML metadata compliant" will not be required to recognise other encoding schemes.

Figure 6 shows the new recommended implementation for mathematical problem type metadata.

RDF containers could be used to indicate that a given CellML element has more than one problem type (this will only be necessary if the problem types are not both members of the same superclass). However, there is no potential ambiguity about the meaning of specifying multiple mathematical problem types for a CellML element. Multiple problem types can only mean that the element contains all of the specified problem types. Therefore, this can be encoded using the simpler method of repeating the `<cmeta:math_problem_type>` element.

4 Annotations

The March 15 meeting minutes[5] indicate that the remaining issues with annotation metadata are the addition of a "validation" type of annotation and determination of how to store information about annotation creators. Furthermore, the implementation of the type qualifier in this metadata needs to be made consistent with the metadata qualified with the Dublin Core qualifiers. The annotation creator issue is deferred to a later set of metadata meeting minutes, which will discuss how to store information about people. Solutions to the other two issues are recommended here.

[4] http://www.cellml.org/private/progress_reports/20010315_meeting_minutes.html

[5] http://www.cellml.org/private/progress_reports/20010315_meeting_minutes.html

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cmeta="http://www.cellml.org/2001/03/metadata#"
    xmlns:dc="http://purl.org/dc/elements/1.0/"
    xmlns:dcq="http://purl.org/dc/qualifiers/1.0/">

  <rdf:Description about="some_element_id">
    <cmeta:bio_entity>
      <rdf:Bag>
        <rdf:li>
          <rdf:Description>
            <dc:title>calmodulin</dc:title>
            <dc:title>
              <rdf:Description>
                <dcq:titleType>alternative</dcq:titleType>
                <rdf:value>CaM</rdf:value>
              </rdf:Description>
            </dc:title>
            <cmeta:identifier>
              <rdf:Description>
                <cmeta:identifier_scheme>SWISS-PROT</cmeta:identifier_scheme>
                <rdf:value>CALM_HUMAN</rdf:value>
              </rdf:Description>
            </cmeta:identifier>
          </rdf:Description>
        </rdf:li>
        <rdf:li>
          <rdf:Description>
            <dc:title>troponin C</dc:title>
          </rdf:Description>
        </rdf:li>
        <rdf:li>
          <rdf:Description>
            <cmeta:identifier>
              <rdf:Description>
                <cmeta:identifier_scheme>SWISS-PROT</cmeta:identifier_scheme>
                <rdf:value>CALL_HUMAN</rdf:value>
              </rdf:Description>
            </cmeta:identifier>
          </rdf:Description>
        </rdf:li>
      </rdf:Bag>
    </cmeta:bio_entity>
  </rdf:Description>
</rdf:RDF>
```

FIGURE 5: A complex example of the recommended implementation for biological entity metadata. The referenced CellML element represents the following group of proteins: calmodulin, troponin C, and the protein identified by SWISS-PROT entry `CALL_HUMAN`. The calmodulin biological entity has an alternative name and a database entry. The troponin C biological entity is only identified by name. The `CALL_HUMAN` protein is only identified by database reference.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cmeta="http://www.cellml.org/2001/03/metadata#">

  <rdf:Description about="some_element_id">
    <cmeta:math_problem_type>
      <rdf:Description>
        <cmeta:problem_scheme>GAMS</cmeta:problem_scheme>
        <rdf:value>I1a</rdf:value>
      </rdf:Description>
    </cmeta:math_problem_type>
  </rdf:Description>
</rdf:RDF>
```

FIGURE 6: Recommended implementation of the mathematical problem type metadata. The meaning of the value "GAMS" for the encoding scheme will be controlled by the CellML metadata specification.

4.1 Recommendations

Because we are using an extension of the Dublin Core qualifiers scheme, it is a simple matter to add an additional annotation type. We only need to add "validation" to our list of supported values for the `<cmeta:annotation_type>` element.

Figure 7 shows the recommended implementation for annotation metadata.

The CellML metadata specification will define the following values for the `<cmeta:annotation_type>` element. Modellers are free to invent their own values. However, CellML metadata compliant software is not required to recognise any values except the ones defined in the CellML metadata specification.

- **comment**: free-form comment of the person who coded the model into CellML.
- **limitation**: brief description of the limitations/scope of the content of the CellML element
- **modification**: description of a change made to the content of the CellML element
- **validation**: description of the validation of the content of the CellML element. This can be a code. Note that validation codes are unlikely to be interoperable.

5 Additional Dublin Core Elements

The March 15 meeting minutes[6] raised the possibility of including three additional Dublin Core metadata elements in the list of recommended CellML metadata. Note that modellers are always free to use any Dublin Core elements they wish. In fact, modellers are free to use *any* metadata elements that they wish, as long as they encode the metadata in valid RDF. The CellML metadata specification will recommend a set of metadata elements and an implementation of these elements. Software that claims to be "CellML metadata compliant" will be required to correctly use and recognise these metadata elements. The production of a recommended metadata set and implementation helps to ensure interoperability of metadata produced by different processing software. Although it is hoped that CellML processing software will eventually be able to handle arbitrary RDF, we recognise that this would place an unnecessarily high burden on software. The identification of a limited set of useful metadata elements enables software to deal with these metadata elements without addressing the full complexity of RDF.

[6] http://www.cellml.org/private/progress_reports/20010315_meeting_minutes.html

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cmeta="http://www.cellml.org/2001/03/metadata#"
    xmlns:dc="http://purl.org/dc/elements/1.0/"
    xmlns:dcq="http://purl.org/dc/qualifiers/1.0/">

  <rdf:Description about="some_element_id">
    <cmeta:annotation>
      <rdf:Description>
        <cmeta:annotation_type>limitation</cmeta:annotation_type>
        <rdf:value>
        This component is only valid for temperatures above 20 degrees C
        </rdf:value>
        <dc:creator>Betty Smith</dc:creator>
        <dc:date>
          <rdf:Description>
            <dcq:dateScheme>W3C-DTF</dcq:dateScheme>
            <dcq:dateType>created</dcq:dateType>
            <rdf:value>2001-03-28</rdf:value>
          </rdf:Description>
        </dc:date>
      </rdf:Description>
    </cmeta:annotation>
    <cmeta:annotation>
      <rdf:Description>
        <cmeta:annotation_type>validation</cmeta:annotation_type>
        <rdf:value>
        Physiome level 2
        </rdf:value>
        <dc:creator>Betty Smith</dc:creator>
        <dc:date>
          <rdf:Description>
            <dcq:dateScheme>W3C-DTF</dcq:dateScheme>
            <dcq:dateType>created</dcq:dateType>
            <rdf:value>2001-03-28</rdf:value>
          </rdf:Description>
        </dc:date>
      </rdf:Description>
    </cmeta:annotation>
  </rdf:Description>
</rdf:RDF>
```

FIGURE 7: Recommended implementation of annotation metadata. Note that the contents of the **<dc:creator>** element are provided for example only. Recommendations for how CellML metadata should handle information about people will be provided in a separate document.

5.1 Recommendations on Additional Dublin Core Elements

The initial CellML metadata specification will include the **<dc:description>** and **<dc:contributor>** elements, and their qualifiers. The use of the **<dc:relation>** element is more complicated, and is therefore postponed for a later version of the specification.

The **<dc:description>** element is used to store a short text description of a resource. The **<dc:contribut** element is used to indicate that a person contributed to a resource, but did not actually create it (an example of this is an editor).

An example of the implementation of these two Dublin Core elements is given in Figure 8.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.0/"
    xmlns:dcq="http://purl.org/dc/qualifiers/1.0/">

  <rdf:Description about="some_element_id">
    <dc:description>
      <rdf:Description>
        <dcq:descriptionType>abstract</dcq:descriptionType>
        <rdf:value>
        This element uses simple mass-action kinetics to describe the
        A + B <-> C + D reaction.
        </rdf:value>
      </rdf:Description>
    </dc:description>
    <dc:contributor>
    Jane Doe
    </dc:contributor>
  </rdf:Description>
</rdf:RDF>
```

FIGURE 8: Recommended implementation of the Dublin Core description and contributor metadata. Note that the content of the contributor metadata element will follow the same recommendations for handling information about people that are used for creator metadata.